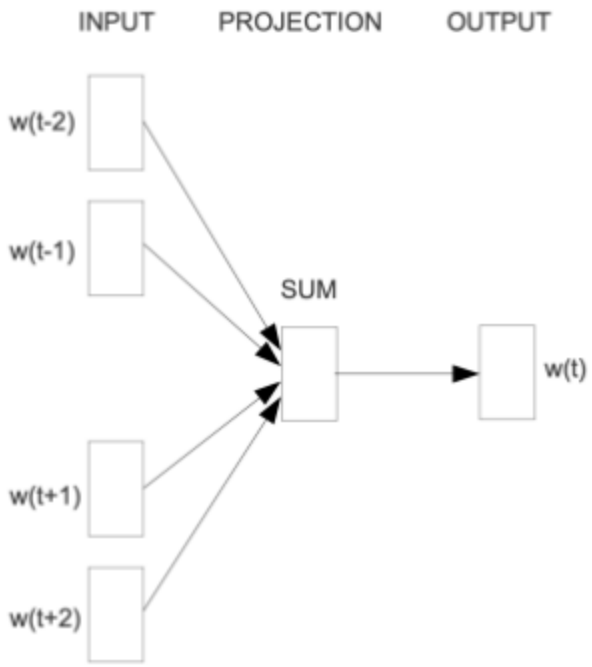
Word2Vec

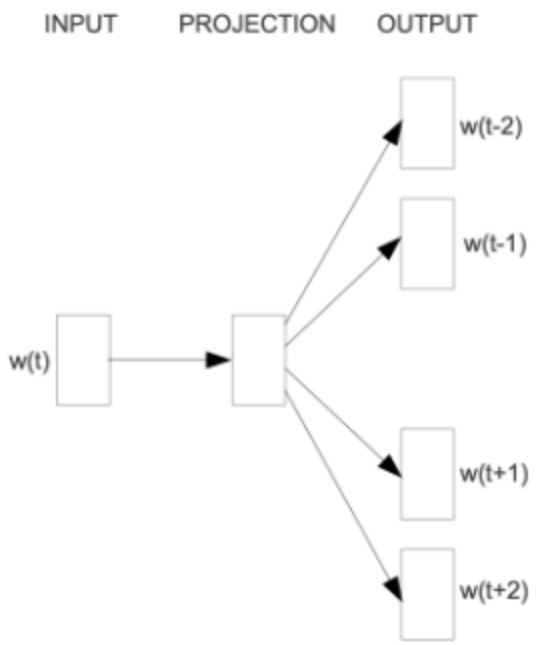
1 CBOW(Countinous Bag-of-words Model)



- 일반적인 NNLM과 유사한 구조를 가짐
- 비선형의 hidden layer가 없음 -> 계산량이 적고 빠름
- 모든 input 단어 벡터들을 average 하여 바로 출력층으로 전달 -> 같은 위치로 projection 됨
- 문맥의 미래와 과거 단어를 모두 사용하여 예측 성능을 향상
- 단어의 순서가 큰 영향을 미치지 못함
  -계산 복잡도

$$Q = N \times D + D \times log_2 V$$

2 Skip-gram(Continuous Skip-gram Model)



- 현재 단어를 input으로 하고 주변 단어들을 예측하는 구조로 학습
- 단어 간의 간격이 멀수록 연관성이 낮으므로 weight를 낮게 조절
  -계산 복잡도(CBOW에 비해 연산량 많음)

$$Q = C \times (D + D \times log_2 V)$$

ELMo

sentence 단위의 word 표현을 학습하기 위해 Bidirectional LSTM을 기반으로 한 언어 모델
주어진 텍스트에서 앞뒤 문맥을 모두 고려하여 단어의 의미를 이해

forward LSTM : 현재 토큰을 기준으로 다음 토큰이 나올 확률을 예측

$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_1, t_2, \ldots, t_{k-1}).$$

backward LSTM : 현재 토큰을 기준으로 이전 토큰이 나올 확률을 예측

$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_{k+1}, t_{k+2}, \ldots, t_N).$$

-> 두 방향의 Log likelihood를 최대화 하는 방식으로 학습. 이때, 가중치는 서로 공유하는 방식으로 학습

$$\sum_{k=1}^{N} \left(\log\, p(t_k|t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log\, p(t_k|t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \right)$$

ELMo는 biLM에서 등장하는 중간 매체 layer의 표현을 합침. biLM의 L개의 layer는 각 토큰 당 2L+1 개의 표현 학습 -> 모든 layer를 하나의 벡터로 압축
-> 이를 통해 얻은 임베딩은 기존 임베딩의 Input에 concat하거나 output에 concat하여 활용